APPLICATION

FOR

UNITED STATES LETTERS PATENT

APPLICANT(S) NAME: B. K. Wade et al

TITLE: Method And Computer System For Controlling Access By Applications To This And Other Computer Systems

DOCKET NO. END9-1999-0107

INTERNATIONAL BUSINESS MACHINES CORPORATION

Certificate of Mailing Under 37 CFR 1.10

I hereby certify that, on the date shown below, this correspondence is being deposited with the United States Postal Service in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C., 20231 as "Express Mail Post Office to Addressee".

"Express Mail" Label Number EL172582484

On September 20, 2000

Denise Jurik

Typed or Printed Name of Person Mailing Correspondence

Denise M Jurik 9/20/00

Signature of Person Mailing Correspondence *Date*

METHOD AND COMPUTER SYSTEM FOR CONTROLLING ACCESS BY APPLICATIONS TO THIS AND OTHER COMPUTER SYSTEMS

FIELD OF THE INVENTION

The present invention generally relates to computer systems and associated methods and more particularly to a system and associated method for protecting computer systems from unauthorized access by certain applications.

DESCRIPTION OF THE RELATED ART

With the increasing sophistication of computer networks and the ability of the public to access private networks (e.g., through the Internet) there have been a number of recent systems and computer programs developed for protecting private networks from being improperly accessed by the public. For example, it is common to use sophisticated passwords and other security devices to prevent unauthorized access to a private network. However, the sophistication of the password and the ability to break passwords is an ever escalating "competition" which quickly renders password security applications obsolete. Therefore, password security applications require constant updating and maintenance. Further, all of the users must have a password to access the system, which limits the effectiveness of password security for Web servers.

Another such security measure is known as a firewall. In one type of firewall system, the users transfer files to a firewall host, and then log into the firewall and subsequently transfer the file to the desired external party. Using a firewall prevents users from making a direct connection outside the private network and also prevents the public from reaching the private network because information can only be transferred by the firewall host. Other systems allow certain external users to "tunnel" through the firewall to allow direct access to the private network. Such systems contain high levels of security to allow the tunneling process to take

place. However, firewalls and tunnels through firewalls are very cumbersome because the private network and the public need to deal with the intermediary firewall. This is a slow and cumbersome process and adds to the cost and complexity of the system.

Other systems use a proxy server to hide the identity of the private network. In such a system, communications are passed through the proxy server, which appears to the public to be the private network. However, only the proxy knows the true identity and location of the private network and the proxy does not allow the public access to the private network. As with the firewall system, the proxy system is cumbersome, expensive and requires constant maintenance.

Therefore, there is a need for a low-overhead system which limits access by an untrusted computer system to a private trusted computer system, but which simultaneously allows the trusted system full access to the untrusted system.

SUMMARY OF THE INVENTION

According to the present invention, application execution contexts within an untrusted computer system are classified as either trusted or untrusted based on distinctive application execution context names. A human administrator of the untrusted system assigns these distinctive application execution context names. The applications in the untrusted system cannot change the names of their own execution contexts (or preferably any other execution contexts in the untrusted computer system).

The untrusted application execution contexts are fenced off from certain parts of the untrusted computer system such that the untrusted application execution contexts cannot interrogate or change critical system data of the untrusted computer system.

Only applications on the untrusted computer system which execute in trusted execution contexts can initiate a connection with a trusted computer system. The trusted computer system can initiate connections with any execution context on the untrusted computer system. Only the untrusted application execution contexts on the untrusted system can initiate connections with an external computer system. The connections originating on the external system can terminate only at the untrusted system and can terminate only at untrusted execution contexts therein.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects, aspects and advantages will be better understood from the following detailed description of preferred embodiments of the invention with reference to the drawings, in which:

Figure 1 is a schematic diagram of the present invention.

Figure 2 is a flow diagram illustrating operation of the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS OF THE INVENTION

Referring now to the drawings, and more particularly to Figure 1, a first embodiment of the present invention is illustrated in schematic form. The trusted system 10 shown in Figure 1 comprises a private network to which users are permitted access through some form of security such as passwords, physical restriction, etc. The external system 12 shown in Figure 1 represents computer networks owned by the public, such as the Internet and other private networks. One major concern is that the external system 12 may try to reach the trusted computer system 10 to misappropriate data or destroy the trusted system 10.

To prevent this, the invention positions an untrusted computer system 11 between the trusted computer system 10 and the external system 12. The untrusted system 11 includes an operating system 14 which controls many devices such as storage device 15 (e.g., a disk drive). The untrusted system exists to host data or run applications which must be made available to external system 12.

A key feature of the invention is that the untrusted system includes some applications that run in trusted application execution contexts 13 and other applications that run in untrusted application execution contexts 16. With the invention, the untrusted application execution contexts 16 cannot initiate communications with the trusted system 10. However, trusted application execution contexts 13 can initiate connections with the trusted system 10. Programs running on trusted system 10 can initiate connections to any context on untrusted system 11.

The invention distinguishes between trusted application execution contexts 13 and untrusted application execution contexts 16 according to distinctive application execution context names. A human administrator of system 11 with operating system 14 assigns each application execution context a distinctive name, and an application running in untrusted system 11 cannot tamper with the name of any execution context (i.e., its own or any others). The creation of the distinctive names and the classification of the names as trusted or untrusted is performed with a control in the operating system 14. In one example, the prefix for the trusted names (user ids) is a specifically-chosen alphanumeric character string, (i.e., the administrator of the operating system might specify that all names beginning with the characters "SFS" are trusted).

Certain applications running on behalf of users on external system 12 are given names by a human administrator of the operating system 14 which indicate that they are running in untrusted execution contexts 16. Other applications running on behalf of trusted system 10 are given other names by the human administrator of operating system 14 which indicate that they are running in trusted execution contexts 13.

Any application which is running in a context that has been assigned a "trusted" name by the system administrator executes as a trusted application. These "trusted" contexts (user ids) are unable to communicate with the external system 12. By segregating the application contexts into trusted and untrusted, allowing only the trusted application contexts 13 to initiate connections with the trusted system 10, and allowing the external network 12 to access only the untrusted contexts 16, security is achieved.

In a preferred embodiment two distinct communications protocols are used to distinguish between requests from the external system 12 and the trusted system 10. Requests from the external system 12 use one protocol (e.g., the TCP/IP protocol). Communications to/from the trusted system 10 use a different protocol (e.g., the APPC communications protocol). In this embodiment, the operating system 14 and the storage 15 are set up such that they cannot be manipulated or administered via the TCP/IP protocol (that is, TELNET access is disabled). Therefore, the operating system 14 and storage 15 cannot be compromised by users on external system 12. However, distinct communications protocols are not required to make the invention operable. So long as the operating system can determine the origin of a request through some means (e.g., by noting the TCP/IP interface that the request came through) the invention will operate properly.

Therefore, the invention provides a method for connecting two environments where there is a trusted side 10 and an untrusted side 11 so that the trusted side 10 can have access to all of the data (i.e., the trusted network 10 itself and the untrusted system 11), but the external system 12 can access only a subset of such data (i.e., the data made available through the applications running in untrusted contexts 16).

For example, the invention is very useful where a business has an Internet web page (which would reside on the untrusted system 11) yet still wants to have the Internet web data (in addition to all of its internal business data) accessible by employees who are on the trusted

system 10. In the foregoing situation, the business with the trusted system 10 would need full control over access to the trusted system 10, through conventional security measures (e.g., passwords, physical isolation, etc.).

The security of the invention is achieved by the inventive ability of the operating system 14 to decide which application execution contexts are permitted to initiate connections to the trusted system 10. The communications software on the operating system 14 is programmed so that connection attempts with the trusted system 10 cannot originate from the untrusted application execution contexts 16. As mentioned above, the operating system 14 lets connection attempts to the trusted system 10 originate only from trusted execution contexts 13. The untrusted computer system 11 has a communication software stack and operating system whose controls and configuration parameters are adequately fenced from potentially untrustable applications. In other words, the operating system and communication stack are installed and configured such that untrusted contexts do not have access to their configuration information and are not authorized to control them. This is typically done with file permissions but each operating system will have different means.

With the system described above, if a user on the trusted system 10 attempts to connect to the untrusted system 11 to access some data, the user would be communicating with a trusted application execution context 13 and the connection would be allowed (and data would flow over the connection). However, if an external user 12 (accessing untrusted system 11 via an untrusted application execution context 16) tries to initiate a connection with the trusted system 10, the connection is rejected by the operating system 14. Therefore, the invention allows connections between the trusted system 10 and the untrusted system 11 to be initiated only from the trusted system 10, or from an application running in a trusted execution context 13.

Figure 2 is a flow chart illustrating such operation of the present invention. In step 20, operating system 14 determines the name of the context originating a communication. This determination is made by retrieving the contents of the "context name" field of the data structure

(a.k.a. "control block") describing the initiating context. Next, in step 21, the operating system 14 compares the context name to a list of trusted names (maintained in primary memory or in storage 15) or to a convention for trusted names (such as a particular prefix indicating trustworthiness). Then, operating system 14 determines whether the application execution context that is originating communication is trusted or untrusted. If the application execution context is untrusted, the context is not permitted to initiate communication with trusted system 10, as shown in step 22. However, if the application execution context is trusted, the context is permitted to initiate communication with the trusted system 10, as shown in step 23.

As an example, programming implementing steps 20-23 of the present invention could be added to an existing IBM VM/ESA Version 2 Release 4 operating system to yield operating system 14. In this scenario, untrusted system 11 would run the VM/ESA operating system augmented with the present invention and trusted system 10 would run the standard VM/ESA operating system without such augmentation. A networking connection between these two VM/ESA operating systems could be an existing VM/ESA Inter-System Facility for Communication (ISFC) link. When the augmented VM/ESA operating system 14 in untrusted system 11 determines that a connection is not permitted with trusted system 10 because the connection request originates from an untrusted execution context 16, operating system 14 does not let the untrusted context 16 use ISFC. However, if operating system 14 determines that a connection is permitted with trusted system 10 because the connection request originates from a trusted execution context 13, operating system 14 lets the trusted execution context 13 use ISFC to initiate Advanced Program-to-Program Communication (APPC) connection to the trusted system 10. The APPC protocol is described in IBM publication SC24-5760, "VM/ESA CP Programming Services", and other references.

By allowing trusted application execution contexts 13 in the untrusted system 11 to initiate communications with the trusted system 10, various VM/ESA features requiring such capability, such as the Shared File System (SFS), are easily enabled. The present invention would permit SFS to initiate the connection to trusted system 10 because SFS would be running in a trusted execution context 13.

With the present invention, system administration and overhead are reduced because only one operating system on two (untrusted and trusted) computer systems are required (even though one is modified). As explained above in the "Description of the Related Art", in order to conventionally attain the results produced by the present invention, several servers running multiple conventional operating systems would have to be utilized to ensure that users operating the untrusted system were denied access to the trusted system. Each of these servers would require more code (which slows the system down) and additional administrator time to set up the servers. Further, such a conventional system would be substantially more complicated to maintain than the present invention.

As noted above, applications running in untrusted execution contexts 16 cannot interrogate or update critical or sensitive system 11 data, such as a password file, the set of enrolled users, the list of trusted execution context names or the record of the name prefix that distinguishes trusted contexts from untrusted ones. This can be accomplished based on file permission and access schemes specific to the type of operating system 14. For example, when operating system 14 is an augmented VM/ESA operating system as explained above, applications running in untrusted execution contexts 16 are denied knowledge of and access to the disk volumes containing the CP directory, the TCP/IP communication stack configuration files, and the startup script containing the commands that activate ISFC filtering and define the trusted context name prefix. The CP directory contains the list of known users (that is, execution contexts) and their passwords. The TCP/IP communication stack configuration files list the names of the contexts that are allowed to communicate with external system 12. The startup script, run automatically by VM/ESA at startup time, contains privileged commands that activate

the filtering of ISFC activity based on context name and define the name prefix that identifies trusted execution contexts.

While the invention has been described in terms of preferred embodiments, those skilled in the art will recognize that the invention can be practiced with modification within the spirit and scope of the appended claims.